CLAIMS

I claim:

1. A clock processing logic for determining an edge of a clock signal indicated in a sample vector by a bit location corresponding to a transition from one or more bits of a first value on one side of said bit location to one or more bits of a second value on another side of said bit location, wherein said bit location varies from cycle to cycle according to reference voltage and temperature variations affecting said clock signal, comprising:

edge detection logic configured to compare adjacent pairs of bits of said sample vector starting from one end of said sample vector to another end of said sample vector until a bit location corresponding to a transition from one or more bits of a first value on one side of said bit location to one or more bits of a second value on another side of said bit location is detected; and

sensitivity adjustment logic configured to adjust said bit location according to information of at least one other bit location corresponding to a previous cycle of said clock signal that was previously detected by said edge detection logic.

2. The clock processing logic according to claim 1, wherein a prior bit location was previously detected by said edge detection logic during a prior cycle of said clock signal, and said sensitivity adjustment logic

is configured to adjust said bit location according to the following logic:

if said bit location is less than said prior bit location by a predefined number of bit locations, then said bit location is adjusted to be said prior bit location decremented by one bit location; else

if said bit location is greater than said prior bit location by said predefined number of bit locations, then said bit location is adjusted to be said prior bit location incremented by one bit location; else

said bit location is adjusted to be said prior bit location.

3. The clock processing logic according to claim 1, wherein prior bit locations have been previously detected by said edge detection logic during prior cycles of said clock signal, and said sensitivity adjustment logic is configured to adjust said bit location according to a moving average of a predefined number of said prior bit locations.

4. The clock processing logic according to claim 1, further comprising a sensitivity counter incremented each cycle of said clock signal, wherein a prior bit location was previously detected by said edge detection logic during a prior cycle of said clock signal, and said sensitivity adjustment logic is configured to adjust said bit location according to the following logic:

if said bit location is less than said prior bit location by a predefined number of bits and said prior bit location is greater than said predefined number of bits from a first bit location of said sample vector, then said bit location is adjusted to said prior bit location decremented by one bit location; else

if said bit location is greater than said prior bit location by said predefined number of bits and said prior bit location is greater than said predefined number of bits from a last bit location of said sample vector, then said bit location is adjusted to said prior bit location incremented by one bit location; else

if said sensitivity counter has not counted to a predefined sensitivity period count, then said bit location is adjusted to said prior bit location; else

if said sensitivity counter has counted to said predefined sensitivity period count, then said sensitivity counter is reset and if said bit location is less than said prior bit location and said prior bit location is not said first bit location of said sample vector, then said bit location is adjusted to said prior bit location decremented by one bit location; else

if said sensitivity counter has counted to said predefined sensitivity period count, then said sensitivity counter is reset and if said bit location is greater than said prior bit location and said prior bit location is not said last bit location of said sample vector, then said bit location is adjusted to said prior bit location incremented by one bit location; else

said bit location is adjusted to said prior bit location.

5. The clock processing logic according to claim 1, further comprising a metastability filter configured to correct errors in said sample vector due to metastability problems when capturing said sample vector, wherein said metastability filter processes said sample vector before providing said sample vector to said edge detection logic.

6. The clock processing logic according to claim 5, wherein said metastability filter is configured to filter said sample vector for each bit location of said sample vector between high and low boundary bit locations so that the bit location being filtered is set to a value that is the predominant value of all bit locations between a low end filter bit location equal to the bit location less a predefined number of bit locations and a high end filter bit location equal to the bit location plus said predefined number of bit locations.

7. The clock processing logic according to claim 6, wherein said high boundary bit location is equal to a highest bit location in said sample vector less said predefined number of bit locations, and said low boundary bit location is equal to a lowest bit location of said sample vector plus said predefined number of bit locations.

8. The clock processing logic according to claim 5, wherein said metastability filter is configured to filter said sample vector for each bit location of said sample vector between high and low boundary bit locations according to the following logic:

if a filter size is equal to three, and if a sum of values of the bit location being filtered and bit locations one less and one more than the bit location being filtered is greater than "1", then the bit location being filtered is set to "1"; else

if said filter size is equal to three, and if said sum of said values of the bit location being filtered and said bit locations one less and one more than the bit location being filtered is not greater than "1", then the bit location being filtered is set to "0".

9. The clock processing logic according to claim 1, wherein said first value used in said edge detection logic is a "1" and said second value used in said edge detection logic is a "0".

10. The clock processing logic according to claim 9, wherein said edge detection logic finds a first occurrence of said transition in said sample vector.

11. The clock processing logic according to claim 1, further comprising an error adjustment logic configured to adjust said bit location detected by said edge detection logic by a number of bit locations as

specified in a field of a control register readable by said error adjustment logic.

12. The clock processing logic according to claim 1, further comprising a manual override logic configured to provide a user specified bit location indicating an edge of said clock signal when said clock processing logic is not enabled except for said manual override logic.

13. The clock processing logic according to claim 1, wherein said edge detection logic and said sensitivity adjustment logic are implemented by hardwired logic circuits.

14. The clock processing logic according to claim 1, wherein said edge detection logic and said sensitivity adjustment logic are implemented by a processing unit.

15. The clock processing logic according to claim 1, wherein said edge detection logic and said sensitivity adjustment logic are implemented by a combination of hardwired logic circuits and a processing unit.

16. A method for processing a sample vector indicating an edge of a clock signal by a bit location corresponding to a transition from one or more bits of a

first value on one side of said bit location to one or more bits of a second value on another side of said bit location, wherein said bit location varies from cycle to cycle according to reference voltage and temperature variations affecting said clock signal, comprising:

detecting a bit location corresponding to a transition from one or more bits of a first value on one side of said bit location to one or more bits of a second value on another side of said bit location; and

adjusting said bit location according to information of at least one other bit location corresponding to a previous cycle of said clock signal that was previously detected.

17. The method according to claim 16, wherein a prior bit location was previously detected during a prior cycle of said clock signal, and said adjusting said bit location comprises:

adjusting said bit location to be said prior bit location decremented by one bit location if said bit location is less than said prior bit location by a predefined number of bit locations; and

adjusting said bit location to be said prior bit location incremented by one bit location if said bit location is greater than said prior bit location by said predefined number of bit locations.

18. The method according to claim 16, wherein prior bit locations have been previously detected during

prior cycles of said clock signal, and said adjusting said bit location comprises adjusting said bit location according to a moving average of a predefined number of said prior bit locations.

19. The method according to claim 16, wherein a prior bit location was previously detected by said edge detection logic during a prior cycle of said clock signal, and said adjusting said bit location comprises:

adjusting said bit location to said prior bit location decremented by one bit location if said bit location is less than said prior bit location by a predefined number of bits and said prior bit location is greater than said predefined number of bits from a first bit location of said sample vector;

adjusting said bit location to said prior bit location incremented by one bit location if said bit location is greater than said prior bit location by said predefined number of bits and said prior bit location is greater than said predefined number of bits from a last bit location of said sample vector;

adjusting said bit location to said prior bit location if a sensitivity counter incremented each cycle of said clock signal has not counted to a predefined sensitivity period count;

adjusting said bit location to said prior bit location decremented by one bit location and resetting said sensitivity counter if said sensitivity counter has counted to said predefined sensitivity period count, said bit location is less than said prior bit location and said

prior bit location is not said first bit location of said sample vector; and

adjusting said bit location to said prior bit location incremented by one bit location and resetting said sensitivity counter if said sensitivity counter has counted to said predefined sensitivity period count, said bit location is greater than said prior bit location and said prior bit location is not said last bit location of said sample vector.

20. The method according to claim 16, further comprising filtering said sample vector for each bit location of said sample vector between high and low boundary bit locations so that the bit location being filtered is set to a value that is the predominant value of all bit locations between a low end filter bit location equal to the bit location less a predefined number of bit locations and a high end filter bit location equal to the bit location plus said predefined number of bit locations.

21. The method according to claim 20, wherein said high boundary bit location is equal to a highest bit location in said sample vector less said predefined number of bit locations, and said low boundary bit location is equal to a lowest bit location of said sample vector plus said predefined number of bit locations.

22. The method according to claim 16, further comprising filtering said sample vector for each bit

location of said sample vector between high and low boundary bit locations by:

setting a bit location being filtered to "1" if a sum of values of the bit location being filtered and bit locations one less and one more than the bit location being filtered is greater than "1"; and

setting said bit location being filtered to a "0" if said sum of said values of the bit location being filtered and said bit locations one less and one more than the bit location being filtered is not greater than "1".

23. The method according to claim 16, wherein said first value is a "1" and said second value is a "0".

24. The method according to claim 23, wherein said detecting a bit location comprises finding a first occurrence of said transition in said sample vector.

25. The method according to claim 16, further comprising: adjusting said bit location corresponding to said transition by a number of bit locations as specified in a field of a control register.

26. A clock processing logic for determining an average clock period and jitter for a first clock signal characterized by sample vectors taken on a per cycle basis of said first clock signal, wherein individual of said sample vectors indicate at least one edge of said first clock signal by a bit location varying from cycle to cycle

according to reference voltage and temperature variations affecting said first clock signal and corresponding to a transition from one or more bits of a first value on one side of said bit location to one or more bits of a second value on another side of said bit location, comprising:

an edge filter configured to generate filtered sample vectors by marking only bit locations corresponding to edges of said first clock signal as indicated in said sample vectors;

sample accumulation logic configured to generate accumulative sample vectors by logically OR-ing a predefined number of said filtered sample vectors for individual of said accumulative sample vectors; and

clock period and jitter processing logic configured to determine a clock period and jitter on said first clock signal for individual of said accumulative sample vectors.

27. The clock processing logic according to claim 26, further comprising a metastability filter configured to correct errors in said sample vectors due to metastability problems when capturing said sample vectors.

28. The clock processing logic according to claim 26, wherein said edge filter marks only positive edges of said sample vectors, wherein a positive edge is defined as a bit location corresponding to a transition from one or more bits of a first value on one side of said bit location to one or more bits of a second value on another side of said bit location.

29. The clock processing logic according to claim 28, wherein said first value is a "1" and said second value is a "0".

30. The clock processing logic according to claim 29, wherein said edge filter comprises:

an AND gate having a first AND-input, a second AND-input, and an AND-output, wherein said first AND-input is coupled to a bit location of a sample vector, and said AND-output is coupled to a corresponding bit location of said filtered sample vector; and

an exclusive-OR gate having a first XOR-input, a second XOR-input, and an XOR-output, wherein said first XOR-input is coupled to said bit location of said sample vector, said second XOR-input is coupled to a next bit location of said sample vector, and said XOR-output is coupled to said second AND-input.

31. The clock processing logic according to claim 26, wherein said sample accumulation logic comprises:

an accumulation register having a plurality of bit locations equal in number to individual of said sample vectors;

a plurality of OR-gates respectively coupled to corresponding bit locations of said accumulation register such that individual of said plurality of OR-gates have a first OR-input coupled to receive an output from a corresponding bit location of said accumulation register, a

second OR-input coupled to a corresponding bit location of a filtered sample vector, and an OR-output coupled to provide an input to said corresponding bit location of said accumulation register; and

control logic configured to latch said OR-outputs of said plurality of OR-gates into said corresponding bit locations of said accumulation register upon each clock cycle of a second clock signal for a predefined number of clock cycles.

32. The clock processing logic according to claim 31, wherein said sample accumulation logic further comprises an accumulation counter incremented by said second clock signal and indicating to said control logic when said predefined number of clock cycles has been counted.

33. The clock processing logic according to claim 26, wherein said clock period and jitter processing logic comprises:

a shift register configured to receive accumulative sample vectors one at a time from said sample accumulation logic;

first and second registers; and

a state machine configured to cause an accumulative sample vector to be loaded into said shift register upon receiving an indication that said accumulative sample vector is ready to be so loaded; read bits from said shift register at the rate of one bit per

cycle of a second clock signal; store a first count in said second register indicating a number of bits read before a first pair of consecutive bits being read out from said shift register transition from a second value to a first value; and store a second count in said first register indicating a number of additional bits read before a second pair of consecutive bits being read out from said shift register transition from said first value to said second value.

34. The clock processing logic according to claim 33, wherein said clock period and jitter processing logic further comprises control logic configured to determine an average clock period and jitter for said first clock signal from said first and said second counts respectively stored in said second and said first registers.

35. The clock processing logic according to claim 34, wherein said control logic determines said average clock period from a sum of said first count and one-half of said second count, and determines said jitter from said second count.

36. The clock processing logic according to claim 33, further comprising smoothing logic configured to eliminate gaps between markings indicated in individual of said accumulative sample vectors before said individual of said accumulative sample vectors are loaded into said shift register.

37. The clock processing logic according to claim 33, wherein said state machine overwrites said second count with a third count in said first register if another bit having said first value is read out of said shift register after reading out said second pair of consecutive bits and before reading out an additional number of bits equal to a pre-selected fraction of said first count.

38. The clock processing logic according to claim 33, wherein said first clock signal and said second clock signal are the same clock signal.

39. A method for determining an average clock period and jitter for a first clock signal characterized by sample vectors taken on a per cycle basis of said first clock signal, wherein individual of said sample vectors indicate at least one edge of said first clock signal by a bit location varying from cycle to cycle according to reference voltage and temperature variations affecting said first clock signal and corresponding to a transition from one or more bits of a first value on one side of said bit location to one or more bits of a second value on another side of said bit location, comprising:

generating filtered sample vectors by marking only bit locations corresponding to edges of said first clock signal as indicated in said sample vectors;

generating accumulative sample vectors by logically OR-ing a predefined number of said filtered

sample vectors for individual of said accumulative sample vectors; and

determining a clock period and jitter on said first clock signal for individual of said accumulative sample vectors.

40. The method according to claim 39, further comprising correcting errors in said sample vectors due to metastability problems when capturing said sample vectors.

41. The method according to claim 39, wherein said determining a clock period and jitter on said clock signal for individual of said accumulative sample vectors, comprises:

loading an accumulative sample vector into a shift register upon receiving an indication that said accumulative sample vector is ready to be so loaded;

reading bits from said shift register at the rate of one bit per cycle of a second clock signal;

storing a first count in a second register indicating a number of bits read before a first pair of consecutive bits being read out from said shift register transition from a second value to a first value; and

storing a second count in a first register indicating a number of additional bits read before a second pair of consecutive bits being read out from said shift register transition from said first value to said second value.

42. The method according to claim 41, further comprising: determining an average clock period and jitter for said first clock signal from said first and said second counts respectively stored in said second and said first registers.

43. The method according to claim 42, wherein said average clock period is determined from a sum of said first count and one-half of said second count, and said jitter is determined from said second count.

44. The method according to claim 41, further comprising eliminating gaps between markings indicated in individual of said accumulative sample vectors before loading said individual of said accumulative sample vectors into said shift register.

45. The method according to claim 41, further comprising overwriting said second count with a third count in said first register if another bit having said first value is read out of said shift register after reading out said second pair of consecutive bits and before reading out an additional number of bits equal to a pre-selected fraction of said first count.

46. The clock processing logic according to claim 45, wherein said first clock signal and said second clock signal are the same clock signal.